Simple Linear Regression

Step 1: Reading and Understanding the Data

In [43]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [44]:

```
import numpy as np
import pandas as pd
```

In [45]:

```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [46]:

```
CarName=pd.read_csv("D://CarPrice_Assignment.csv")
CarName.head()
```

Out[46]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	enginesize
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	...	130
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	...	130
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	...	152
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	...	109
4	5	2	audi 100ls	gas	std	four	sedan	4wd	front	99.4	...	136

5 rows × 26 columns

In [47]:

```
CarName.shape
```

Out[47]:

```
(205, 26)
```

In [48]:

```
CarName.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   car_ID            205 non-null    int64
 1   symboling         205 non-null    int64
 2   CarName           205 non-null    object
 3   fueltype          205 non-null    object
 4   aspiration        205 non-null    object
 5   doornumber        205 non-null    object
```

```
 6   carbody           205 non-null    object
 7   drivewheel        205 non-null    object
 8   enginelocation    205 non-null    object
 9   wheelbase         205 non-null    float64
 10  carlength         205 non-null    float64
 11  carwidth          205 non-null    float64
 12  carheight         205 non-null    float64
 13  curbweight        205 non-null    int64
 14  enginetype        205 non-null    object
 15  cylindernumber    205 non-null    object
 16  enginesize        205 non-null    int64
 17  fuelsystem        205 non-null    object
 18  boreratio         205 non-null    float64
 19  stroke            205 non-null    float64
 20  compressionratio  205 non-null    float64
 21  horsepower        205 non-null    int64
 22  peakrpm           205 non-null    int64
 23  citympg           205 non-null    int64
 24  highwaympg        205 non-null    int64
 25  price             205 non-null    float64
dtypes: float64(8), int64(8), object(10)
memory usage: 33.7+ KB
```

In [49]:

```
CarName.describe()
```

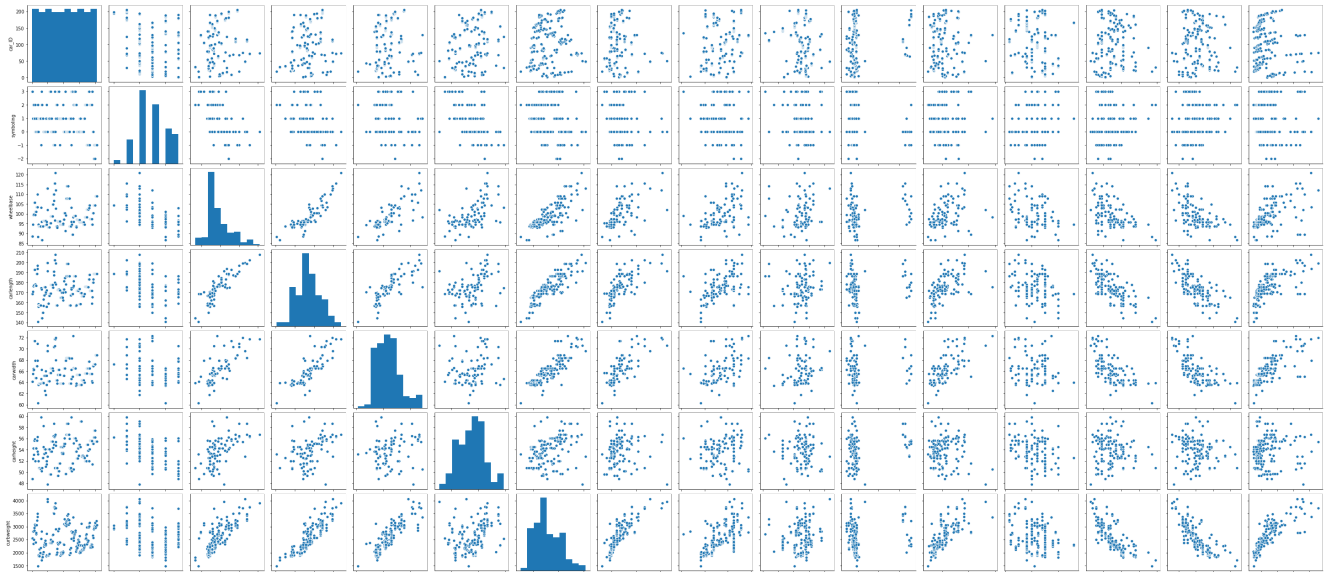
Out[49]:

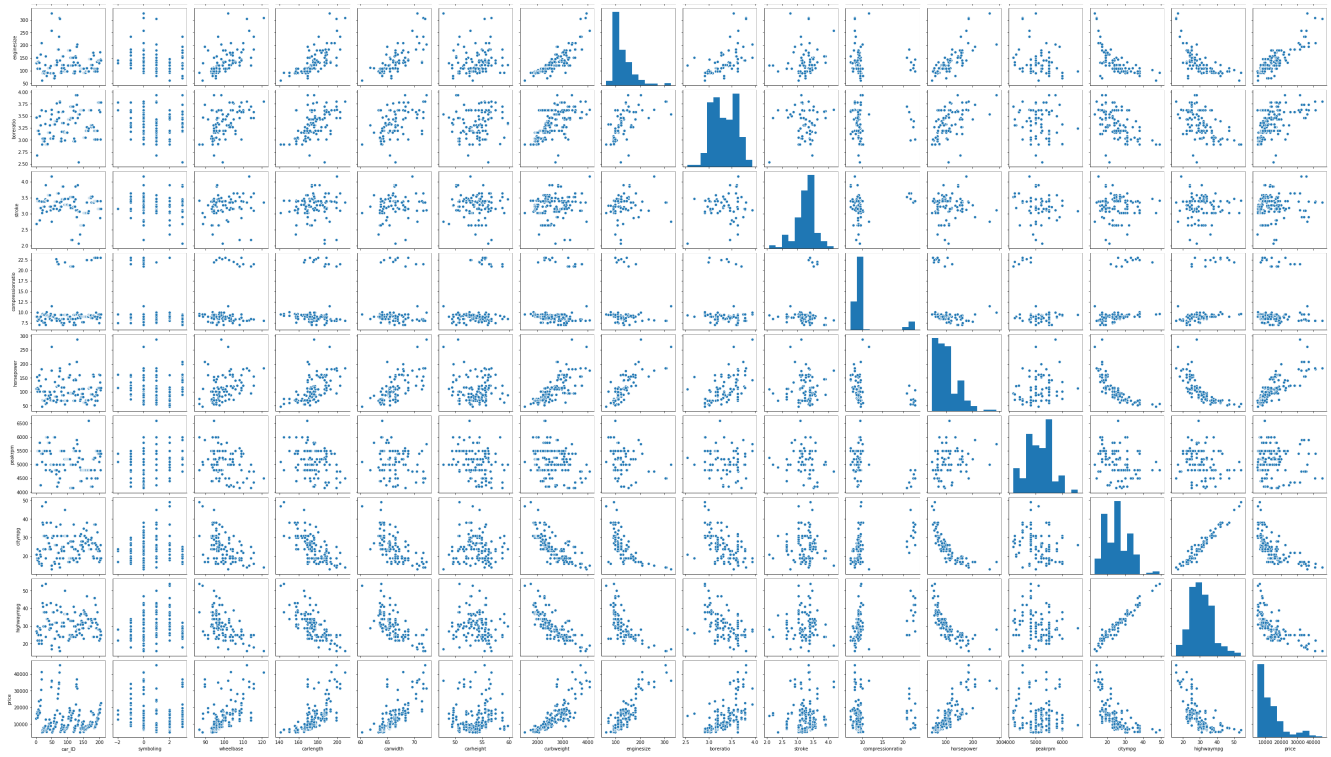
	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight	enginesize	boreratio	stroke	com
count	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	
mean	103.000000	0.834146	98.756585	174.049268	65.907805	53.724878	2555.565854	126.907317	3.329756	3.255415	
std	59.322565	1.245307	6.021776	12.337289	2.145204	2.443522	520.680204	41.642693	0.270844	0.313597	
min	1.000000	-2.000000	86.600000	141.100000	60.300000	47.800000	1488.000000	61.000000	2.540000	2.070000	
25%	52.000000	0.000000	94.500000	166.300000	64.100000	52.000000	2145.000000	97.000000	3.150000	3.110000	
50%	103.000000	1.000000	97.000000	173.200000	65.500000	54.100000	2414.000000	120.000000	3.310000	3.290000	
75%	154.000000	2.000000	102.400000	183.100000	66.900000	55.500000	2935.000000	141.000000	3.580000	3.410000	
max	205.000000	3.000000	120.900000	208.100000	72.300000	59.800000	4066.000000	326.000000	3.940000	4.170000	

Step 2: Visualising the Data

In [50]:

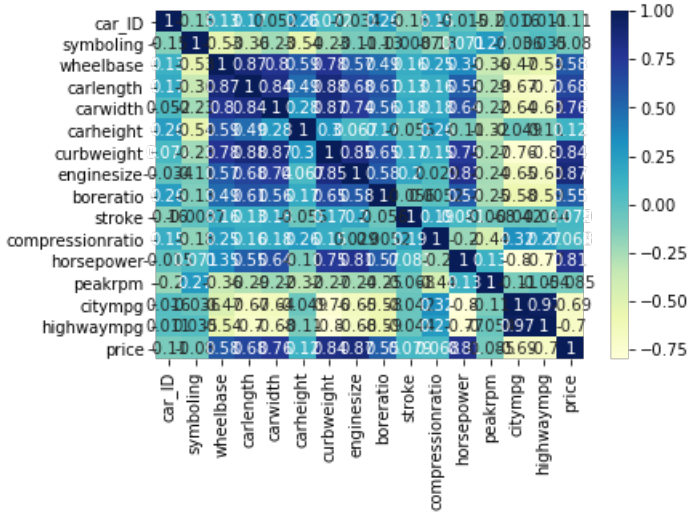
```
sns.pairplot(CarName)
plt.show()
```





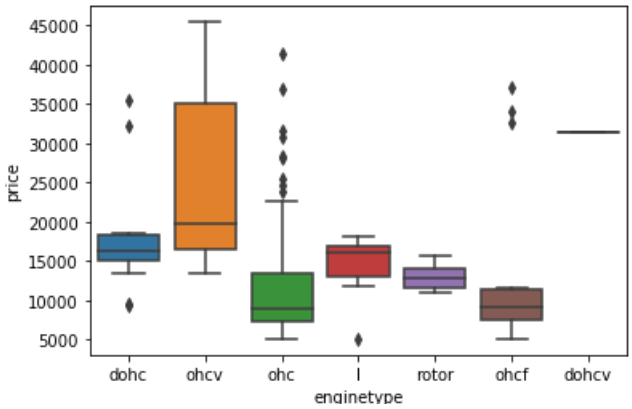
In [51]:

```
sns.heatmap(CarName.corr(), cmap="YlGnBu", annot = True)
plt.show()
```



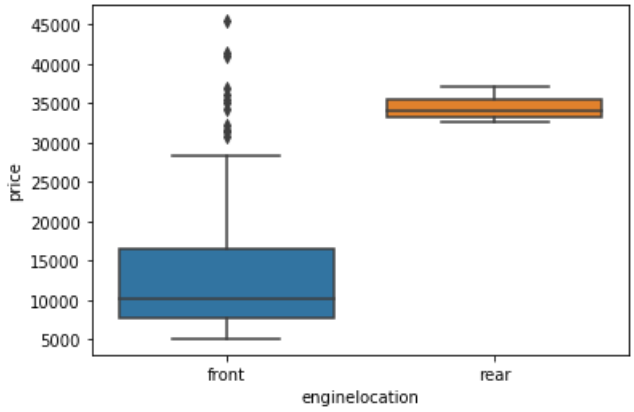
In [52]:

```
sns.boxplot(x='enginetype',y='price',data=CarName)
plt.show()
```



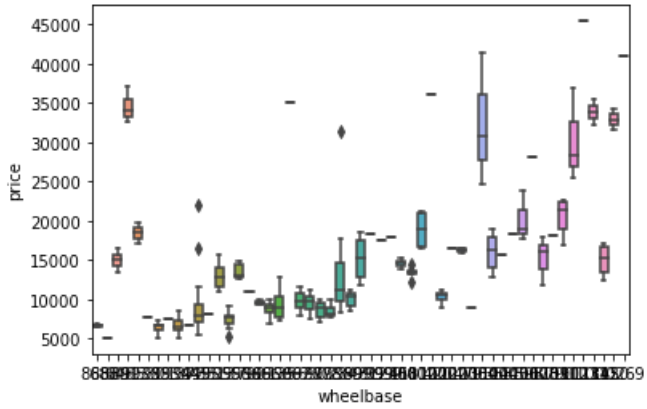
In [53]:

```
sns.boxplot(x='enginelocation',y='price',data=CarName)
plt.show()
```



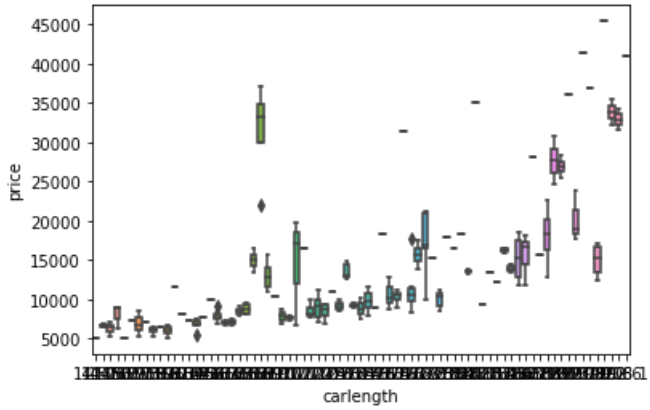
In [54]:

```
sns.boxplot(x='wheelbase',y='price',data=CarName)
plt.show()
```



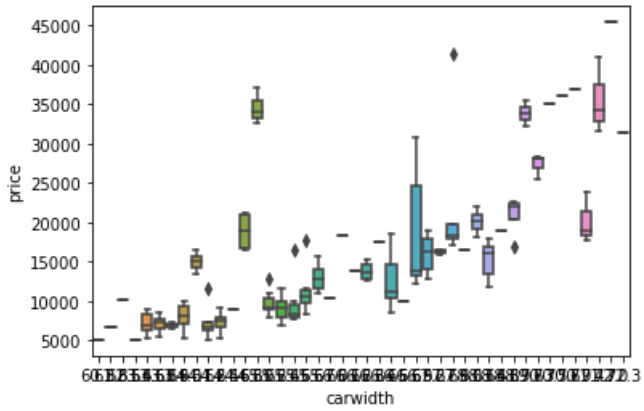
In [55]:

```
sns.boxplot(x='carlength',y='price',data=CarName)
plt.show()
```



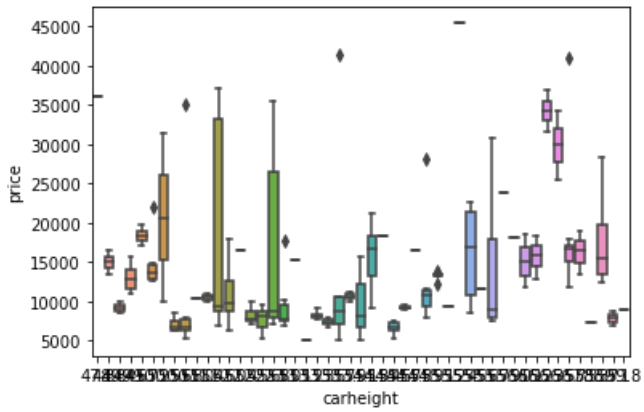
In [56]:

```
sns.boxplot(x='carwidth',y='price',data=CarName)
plt.show()
```



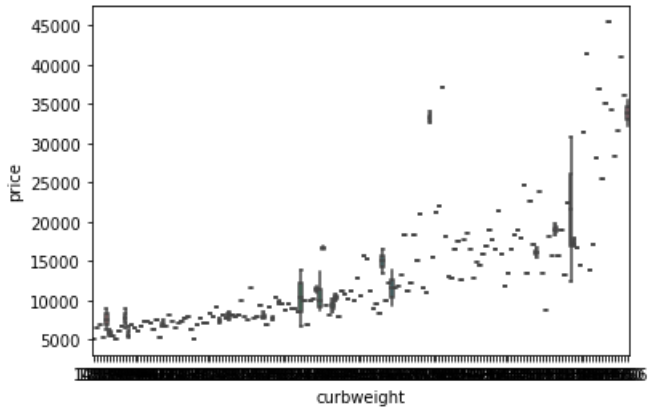
In [57]:

```
sns.boxplot(x='carheight',y='price',data=CarName)
plt.show()
```



In [58]:

```
sns.boxplot(x='curbweight',y='price',data=CarName)
plt.show()
```



In [59]:

```
x=CarName['carlength']
y= CarName['price']
```

In [60]:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size = 0.7, test_size = 0.3,
random_state = 100)
```

In [61]:

In [61]:

```
x_train.head()
```

Out[61]:

```
122    167.3
125    168.9
166    168.7
1      168.8
199    188.8
Name: carlength, dtype: float64
```

In [62]:

```
y_train.head()
```

Out[62]:

```
122     7609.0
125    22018.0
166     9538.0
1      16500.0
199    18950.0
Name: price, dtype: float64
```

In [63]:

```
import statsmodels.api as sm
```

In [64]:

```
x_train_sm = sm.add_constant(x_train)
lr = sm.OLS(y_train,x_train_sm).fit()
```

In [65]:

```
lr.params
```

Out[65]:

```
const       -63647.447004
carlength      442.308944
dtype: float64
```

In [66]:

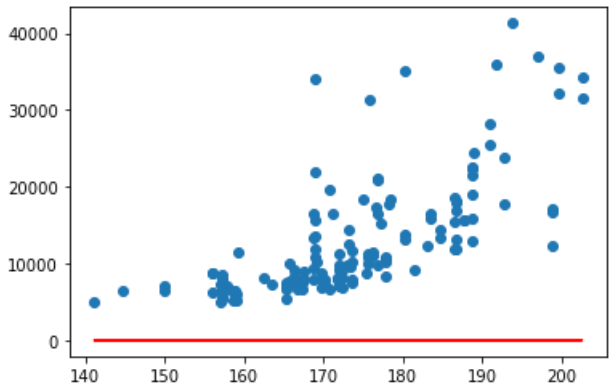
```
print(lr.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.509
Model:                            OLS   Adj. R-squared:                  0.506
Method:                 Least Squares   F-statistic:                     146.4
Date:                Sun, 26 Apr 2020   Prob (F-statistic):           1.46e-23
Time:                        09:08:44   Log-Likelihood:                -1433.2
No. Observations:                 143   AIC:                             2870.
Df Residuals:                     141   BIC:                             2876.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const      -6.365e+04   6355.421    -10.015      0.000   -7.62e+04   -5.11e+04
carlength    442.3089     36.553     12.101      0.000     370.047     514.571
==============================================================================
Omnibus:                       54.186   Durbin-Watson:                   1.898
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              133.614
Skew:                           1.572   Prob(JB):                     9.68e-30
Kurtosis:                       6.541   Cond. No.                     2.41e+03
==============================================================================
```

```
Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.41e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [67]:

```
plt.scatter(x_train, y_train)
plt.plot(x_train, 0.127 + 0.462*x_train, 'r')
plt.show()
```

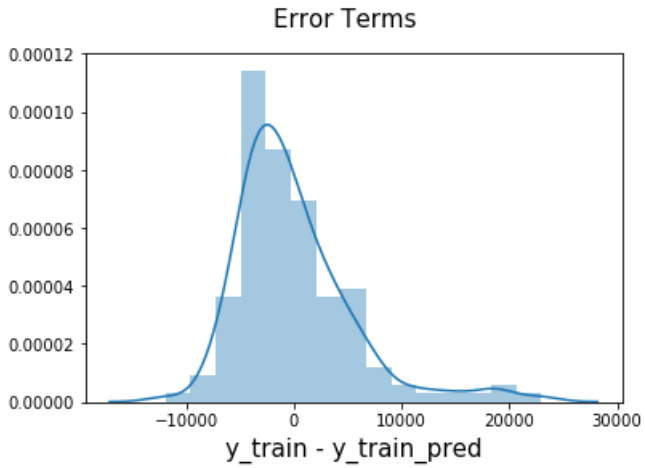


In [68]:

```
y_train_pred = lr.predict(x_train_sm)
res = (y_train - y_train_pred)
```

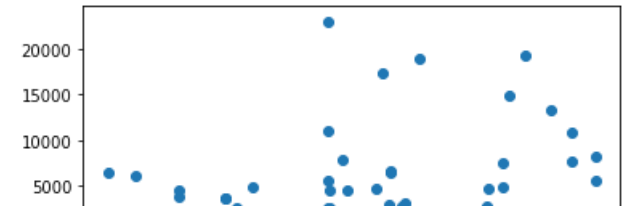
In [69]:

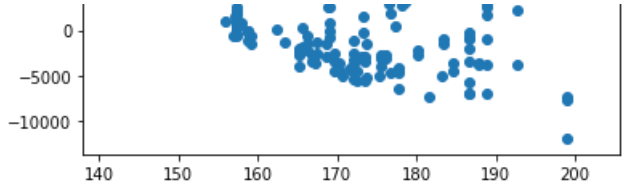
```
fig = plt.figure()
sns.distplot(res, bins = 15)
fig.suptitle('Error Terms', fontsize = 15)
plt.xlabel('y_train - y_train_pred', fontsize = 15)
plt.show()
```



In [70]:

```
plt.scatter(x_train,res)
plt.show()
```





In [71]:

```
x_test_sm = sm.add_constant(x_test)
y_pred = lr.predict(x_test_sm)
```

In [72]:

```
y_pred.head()
```

Out[72]:

```
160     9908.530450
186    12296.998749
59     14995.083310
165    10970.071916
140     5927.749950
dtype: float64
```

In [73]:

```
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
```

In [74]:

```
np.sqrt(mean_squared_error(y_test, y_pred))
```

Out[74]:

```
6602.1821375507725
```

In [75]:

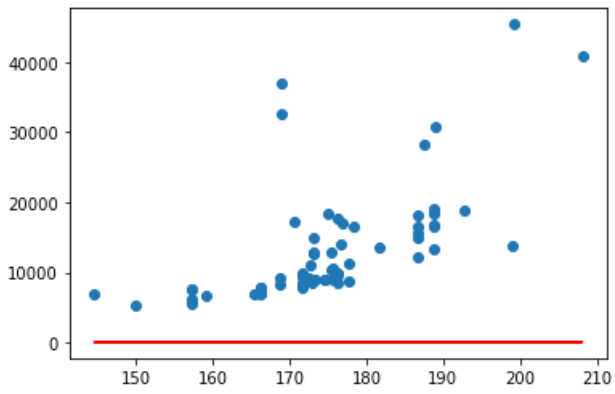
```
r_squared = r2_score(y_test, y_pred)
r_squared
```

Out[75]:

```
0.3775614539285084
```

In [76]:

```
plt.scatter(x_test, y_test)
plt.plot(x_test,0.127 + 0.462* x_test, 'r')
plt.show()
```



In [77]:

```
from sklearn.model_selection import train_test_split
x_train_lm, x_test_lm, y_train_lm, y_test_lm = train_test_split(x, y, train_size = 0.7, test_size =
0.3, random_state = 100)
```

In [78]:

```
x_train_lm.shape
```

Out[78]:

```
(143,)
```

In [79]:

```
x_train_lm
x_train_lm = x_train_lm.values.reshape(-1,1)
x_train_lm
x_test_lm = x_test_lm.values.reshape(-1,1)
```

In [80]:

```
print(x_train_lm.shape)
print(y_train_lm.shape)
print(x_test_lm.shape)
print(y_test_lm.shape)
```

```
(143, 1)
(143,)
(62, 1)
(62,)
```

In [81]:

```
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
lm.fit(x_train_lm, y_train_lm)
```

Out[81]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

In [82]:

```
print(lm.intercept_)
print(lm.coef_)
```

```
-63647.447004327536
[442.3089444]
```

In [83]:

```
corrs = np.corrcoef(x_train, y_train)
print(corrs)
```

```
[[1.         0.71374864]
 [0.71374864 1.        ]]
```

In [84]:

```
corrs[0,1] ** 2
```

Out[84]:

```
0.5094371243649879
```

In [85]:

```
from sklearn.model_selection import train_test_split
x_train,x_test, y_train, y_test = train_test_split(x, y, train_size = 0.7, test_size = 0.3,
random_state = 100)
```

In [86]:

```
from sklearn.preprocessing import StandardScaler, MinMaxScaler
```

In [87]:

```
x_train_scaled = x_train.values.reshape(-1,1)
y_train_scaled = y_train.values.reshape(-1,1)
```

In [88]:

```
x_train_scaled.shape
```

Out[88]:

```
(143, 1)
```

In [89]:

```
scaler = StandardScaler()
x_train_scaled = scaler.fit_transform(x_train_scaled)
y_train_scaled = scaler.fit_transform(y_train_scaled)
```

In [90]:

```
print("mean and sd for x_train_scaled:", np.mean(x_train_scaled), np.std(x_train_scaled))
print("mean and sd for y_train_scaled:", np.mean(y_train_scaled), np.std(y_train_scaled))
```

```
mean and sd for x_train_scaled: 1.6148698540002277e-16 1.0
mean and sd for y_train_scaled: 1.8633113700002627e-16 1.0000000000000002
```

In [91]:

```
x_train_scaled = sm.add_constant(x_train_scaled)
lr_scaled = sm.OLS(y_train_scaled,x_train_scaled).fit()
```

In [92]:

```
lr_scaled.params
```

Out[92]:

```
array([1.68268177e-16, 7.13748642e-01])
```

In [93]:

```
print(lr_scaled.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.509
Model:                            OLS   Adj. R-squared:                  0.506
Method:                 Least Squares   F-statistic:                     146.4
Date:                Sun, 26 Apr 2020   Prob (F-statistic):           1.46e-23
Time:                        09:08:50   Log-Likelihood:                -151.99
No. Observations:                 143   AIC:                             308.0
Df Residuals:                     141   BIC:                             313.9
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
```

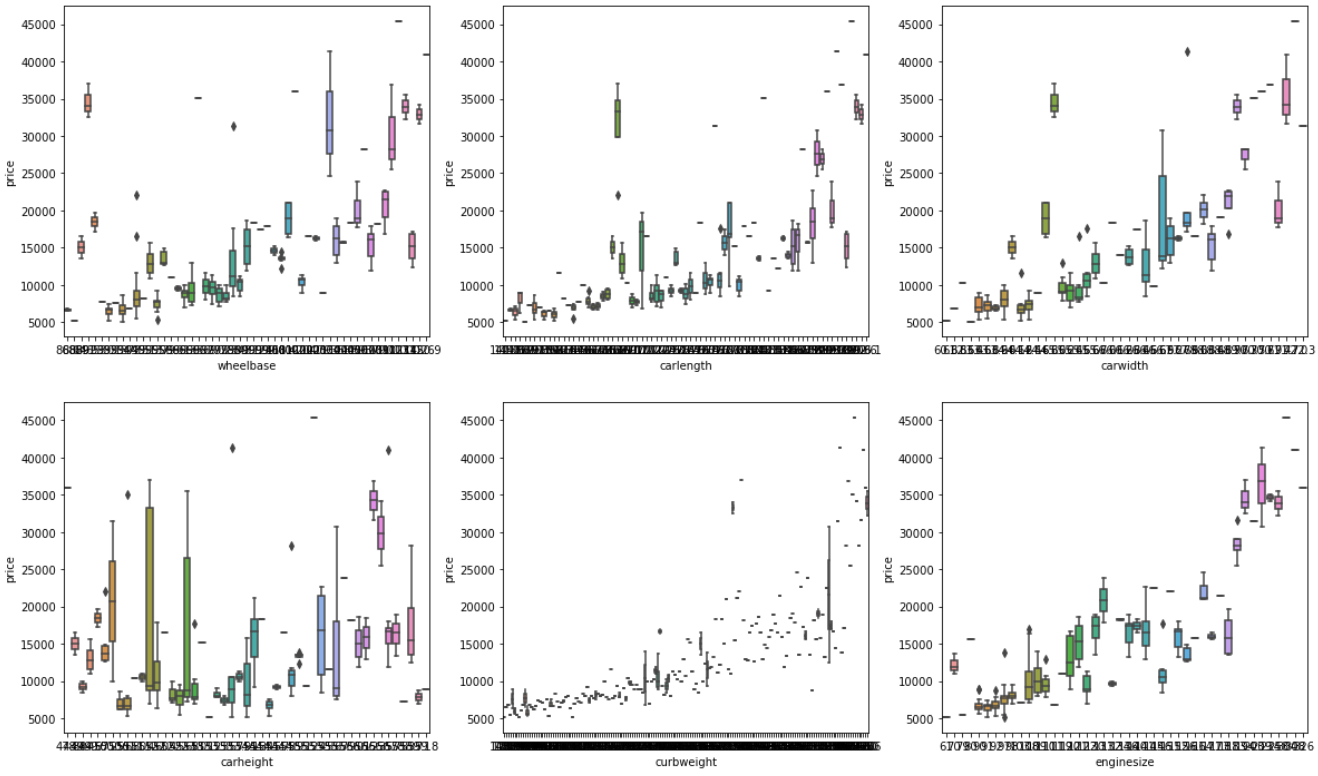
```
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1.683e-16      0.059   2.85e-15      1.000      -0.117       0.117
x1             0.7137      0.059     12.101      0.000       0.597       0.830
==============================================================================
Omnibus:                       54.186   Durbin-Watson:                   1.898
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              133.614
Skew:                           1.572   Prob(JB):                     9.68e-30
Kurtosis:                       6.541   Cond. No.                         1.00
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Multiple Linear Regression

In [94]:

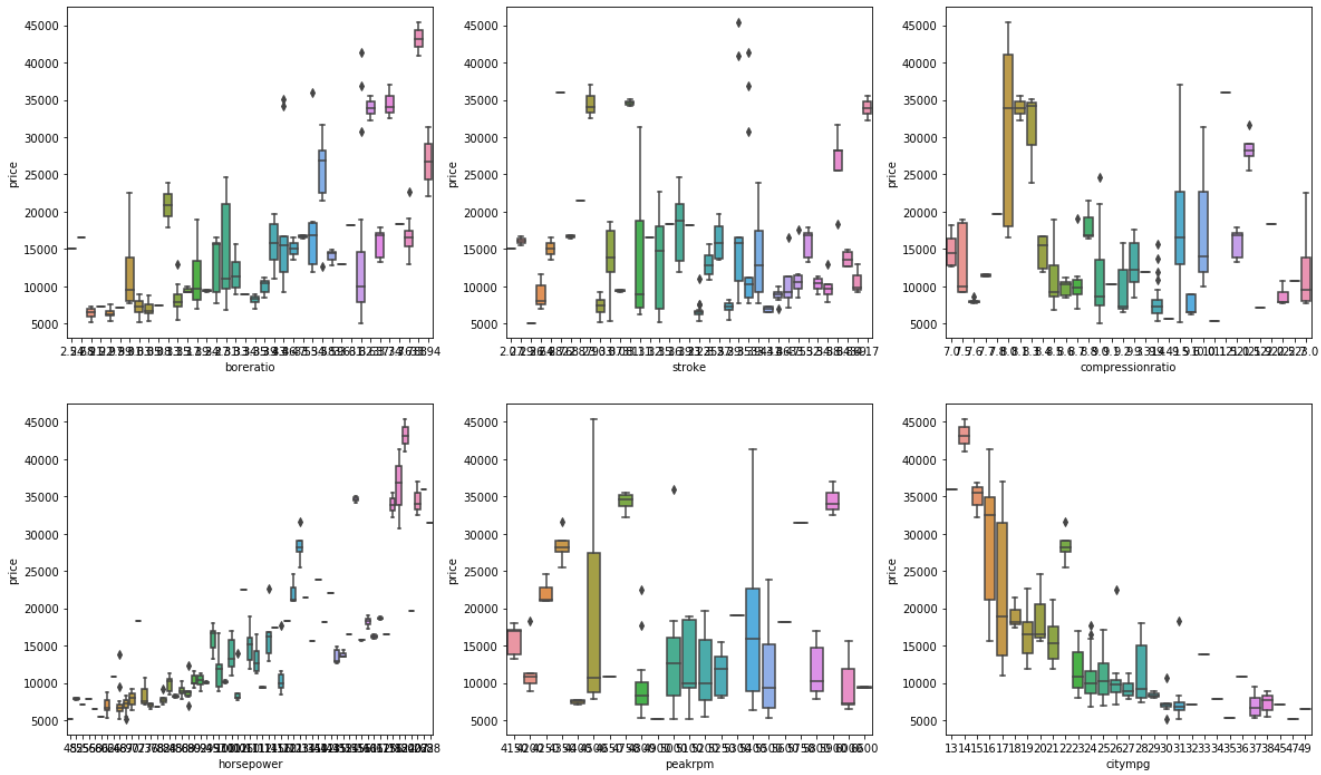
```
plt.figure(figsize=(20, 12))
plt.subplot(2,3,1)
sns.boxplot(x = 'wheelbase', y = 'price', data =CarName)
plt.subplot(2,3,2)
sns.boxplot(x = 'carlength', y = 'price', data =CarName )
plt.subplot(2,3,3)
sns.boxplot(x = 'carwidth', y = 'price', data =CarName )
plt.subplot(2,3,4)
sns.boxplot(x = 'carheight', y = 'price', data = CarName)
plt.subplot(2,3,5)
sns.boxplot(x = 'curbweight', y = 'price', data = CarName)
plt.subplot(2,3,6)
sns.boxplot(x = 'enginesize', y = 'price', data = CarName)
plt.show()
```



In [95]:

```
plt.figure(figsize=(20, 12))
plt.subplot(2,3,1)
sns.boxplot(x = 'boreratio', y = 'price', data = CarName)
plt.subplot(2,3,2)
sns.boxplot(x = 'stroke', y = 'price', data = CarName)
plt.subplot(2,3,3)
sns.boxplot(x = 'compressionratio', y = 'price', data = CarName)
plt.subplot(2,3,4)
```

```
sns.boxplot(x = 'horsepower', y = 'price', data = CarName)
plt.subplot(2,3,5)
sns.boxplot(x = 'peakrpm', y = 'price', data = CarName)
plt.subplot(2,3,6)
sns.boxplot(x = 'citympg', y = 'price', data = CarName)
plt.show()
```



In [96]:

```
status = pd.get_dummies(CarName['cylindernumber'])
```

In [97]:

```
status.head()
```

Out[97]:

	eight	five	four	six	three	twelve	two
0	0	0	1	0	0	0	0
1	0	0	1	0	0	0	0
2	0	0	0	1	0	0	0
3	0	0	1	0	0	0	0
4	0	1	0	0	0	0	0

In [98]:

```
status = pd.get_dummies(CarName['cylindernumber'], drop_first = True)
```

In [99]:

```
CarName = pd.concat([CarName, status], axis = 1)
```

In [100]:

```
CarName.head()
```

Out[100]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	peakrpm
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	...	5000
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	...	5000
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	...	5000
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	...	5500
4	5	2	audi 100ls	gas	std	four	sedan	4wd	front	99.4	...	5500

5 rows × 32 columns

In [101]:

```
CarName.drop(['cylindernumber'], axis = 1, inplace = True)
```

In [102]:

```
CarName.head()
```

Out[102]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	peakrpm
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	...	5000
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	...	5000
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	...	5000
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	...	5500
4	5	2	audi 100ls	gas	std	four	sedan	4wd	front	99.4	...	5500

5 rows × 31 columns

In [103]:

```
status = pd.get_dummies(CarName['CarName'])
status.head()
```

Out[103]:

	Nissan versa	alfa-romero Quadrifoglio	alfa-romero giulia	alfa-romero stelvio	audi 100 ls	audi 100ls	audi 4000	audi 5000	audi 5000s (diesel)	audi fox	...	volkswagen type 3	volvo 144ea	volvo 145e (sw)	volvo 244dl	volvo 245	volvo 246
0	0	0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	0	0	...	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0
3	0	0	0	0	1	0	0	0	0	0	...	0	0	0	0	0	0
4	0	0	0	0	0	1	0	0	0	0	...	0	0	0	0	0	0

5 rows × 147 columns

In [104]:

```
status = pd.get_dummies(CarName['CarName'], drop_first = True)
CarName = pd.concat([CarName, status], axis = 1)
CarName.head()
```

Out[104]:

	car_ID car_ID	symboling symboling	CarName CarName	fueltype fueltype	aspiration aspiration	doornumber doornumber	carbody carbody	drivewheel drivewheel	enginelocation enginelocation	wheelbase wheelbase	...	volkswager volkswager type 3
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	...	0
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	...	0
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	...	0
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	...	0
4	5	2	audi 100ls	gas	std	four	sedan	4wd	front	99.4	...	0

5 rows × 177 columns

In [105]:

```
CarName.drop(['CarName'], axis = 1, inplace = True)
CarName.head()
```

Out[105]:

	car_ID	symboling	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	carlength	...	volkswagen type 3
0	1	3	gas	std	two	convertible	rwd	front	88.6	168.8	...	0
1	2	3	gas	std	two	convertible	rwd	front	88.6	168.8	...	0
2	3	1	gas	std	two	hatchback	rwd	front	94.5	171.2	...	0
3	4	2	gas	std	four	sedan	fwd	front	99.8	176.6	...	0
4	5	2	gas	std	four	sedan	4wd	front	99.4	176.6	...	0

5 rows × 176 columns

In [106]:

```
status = pd.get_dummies(CarName['fueltype'])
status.head()
```

Out[106]:

	diesel	gas
0	0	1
1	0	1
2	0	1
3	0	1
4	0	1

In [107]:

```
status = pd.get_dummies(CarName['fueltype'], drop_first = True)
CarName = pd.concat([CarName, status], axis = 1)
CarName.head()
```

Out[107]:

	car_ID	symboling	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	carlength	...	volvo 144ea	volvo 145e (sw)
0	1	3	gas	std	two	convertible	rwd	front	88.6	168.8	...	0	0
1	2	3	gas	std	two	convertible	rwd	front	88.6	168.8	...	0	0
2	3	1	gas	std	two	hatchback	rwd	front	94.5	171.2	...	0	0
3	4	2	gas	std	four	sedan	fwd	front	99.8	176.6	...	0	0

	car_ID	symboling	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	carlength	...	volvo 144ea	volvo 145e (sw)
4	5	2	gas	std	four	sedan	4wd	front	99.4	176.6	...	0	0

5 rows × 177 columns

In [108]:

```
CarName.drop(['fueltype'], axis = 1, inplace = True)
CarName.head()
```

Out[108]:

	car_ID	symboling	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	carlength	carwidth	...	volvo 144ea	volvo 145e (sw
0	1	3	std	two	convertible	rwd	front	88.6	168.8	64.1	...	0	0
1	2	3	std	two	convertible	rwd	front	88.6	168.8	64.1	...	0	0
2	3	1	std	two	hatchback	rwd	front	94.5	171.2	65.5	...	0	0
3	4	2	std	four	sedan	fwd	front	99.8	176.6	66.2	...	0	0
4	5	2	std	four	sedan	4wd	front	99.4	176.6	66.4	...	0	0

5 rows × 176 columns

In [109]:

```
status = pd.get_dummies(CarName['aspiration'])
status.head()
```

Out[109]:

	std	turbo
0	1	0
1	1	0
2	1	0
3	1	0
4	1	0

In [110]:

```
status = pd.get_dummies(CarName['aspiration'], drop_first = True)
CarName = pd.concat([CarName, status], axis = 1)
CarName.head()
```

Out[110]:

	car_ID	symboling	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	carlength	carwidth	...	volvo 145e (sw)	volvo 244dl
0	1	3	std	two	convertible	rwd	front	88.6	168.8	64.1	...	0	0
1	2	3	std	two	convertible	rwd	front	88.6	168.8	64.1	...	0	0
2	3	1	std	two	hatchback	rwd	front	94.5	171.2	65.5	...	0	0
3	4	2	std	four	sedan	fwd	front	99.8	176.6	66.2	...	0	0
4	5	2	std	four	sedan	4wd	front	99.4	176.6	66.4	...	0	0

5 rows × 177 columns

In [111]:

```
CarName.drop(['aspiration'], axis = 1, inplace = True)
CarName.head()
```

Out[111]:

	car_ID	symboling	doornumber	carbody	drivewheel	enginelocation	wheelbase	carlength	carwidth	carheight	...	volvo 145e (sw)	volvo 244dl
0	1	3	two	convertible	rwd	front	88.6	168.8	64.1	48.8	...	0	0
1	2	3	two	convertible	rwd	front	88.6	168.8	64.1	48.8	...	0	0
2	3	1	two	hatchback	rwd	front	94.5	171.2	65.5	52.4	...	0	0
3	4	2	four	sedan	fwd	front	99.8	176.6	66.2	54.3	...	0	0
4	5	2	four	sedan	4wd	front	99.4	176.6	66.4	54.3	...	0	0

5 rows × 176 columns

In [112]:

```
status = pd.get_dummies(CarName['doornumber'])
status.head()
```

Out[112]:

	four	two
0	0	1
1	0	1
2	0	1
3	1	0
4	1	0

In [113]:

```
status = pd.get_dummies(CarName['doornumber'], drop_first = True)
CarName = pd.concat([CarName, status], axis = 1)
CarName.head()
```

Out[113]:

	car_ID	symboling	doornumber	carbody	drivewheel	enginelocation	wheelbase	carlength	carwidth	carheight	...	volvo 244dl	volvo 245
0	1	3	two	convertible	rwd	front	88.6	168.8	64.1	48.8	...	0	0
1	2	3	two	convertible	rwd	front	88.6	168.8	64.1	48.8	...	0	0
2	3	1	two	hatchback	rwd	front	94.5	171.2	65.5	52.4	...	0	0
3	4	2	four	sedan	fwd	front	99.8	176.6	66.2	54.3	...	0	0
4	5	2	four	sedan	4wd	front	99.4	176.6	66.4	54.3	...	0	0

5 rows × 177 columns

In [114]:

```
CarName.drop(['doornumber'], axis = 1, inplace = True)
CarName.head()
```

Out[114]:

	car_ID	symboling	carbody	drivewheel	enginelocation	wheelbase	carlength	carwidth	carheight	curbweight	...	volvo 244dl	volvo 245
0	1	3	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	...	0	0
1	2	3	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	...	0	0
2	3	1	hatchback	rwd	front	94.5	171.2	65.5	52.4	2823	...	0	0
3	4	2	sedan	fwd	front	99.8	176.6	66.2	54.3	2337	...	0	0
4	5	2	sedan	4wd	front	99.4	176.6	66.4	54.3	2824	...	0	0

5 rows × 176 columns

In [115]:

```
status = pd.get_dummies(CarName['carbody'])
status.head()
```

Out[115]:

	convertible	hardtop	hatchback	sedan	wagon
0	1	0	0	0	0
1	1	0	0	0	0
2	0	0	1	0	0
3	0	0	0	1	0
4	0	0	0	1	0

In [116]:

```
status = pd.get_dummies(CarName['carbody'], drop_first = True)
CarName = pd.concat([CarName, status], axis = 1)
CarName.head()
```

Out[116]:

	car_ID	symboling	carbody	drivewheel	enginelocation	wheelbase	carlength	carwidth	carheight	curbweight	...	volvo diesel	vw dashe
0	1	3	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	...	0	0
1	2	3	convertible	rwd	front	88.6	168.8	64.1	48.8	2548	...	0	0
2	3	1	hatchback	rwd	front	94.5	171.2	65.5	52.4	2823	...	0	0
3	4	2	sedan	fwd	front	99.8	176.6	66.2	54.3	2337	...	0	0
4	5	2	sedan	4wd	front	99.4	176.6	66.4	54.3	2824	...	0	0

5 rows × 180 columns

In [117]:

```
CarName.drop(['carbody'], axis = 1, inplace = True)
CarName.head()
```

Out[117]:

	car_ID	symboling	drivewheel	enginelocation	wheelbase	carlength	carwidth	carheight	curbweight	enginetype	...	volvo diesel	v dashe
0	1	3	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	...	0	
1	2	3	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	...	0	
2	3	1	rwd	front	94.5	171.2	65.5	52.4	2823	ohcv	...	0	
3	4	2	fwd	front	99.8	176.6	66.2	54.3	2337	ohc	...	0	
4	5	2	4wd	front	99.4	176.6	66.4	54.3	2824	ohc	...	0	

5 rows × 179 columns

In [118]:

```
status = pd.get_dummies(CarName['drivewheel'])
status.head()
```

Out[118]:

	4wd	fwd	rwd
0	0	0	1

	4wd	fwd	rwd
0	0	0	1
1	0	0	1
2	0	0	1
3	0	1	0
4	1	0	0

In [119]:

```
status = pd.get_dummies(CarName['drivewheel'], drop_first = True)
CarName = pd.concat([CarName, status], axis = 1)
CarName.head()
```

Out[119]:

	car_ID	symboling	drivewheel	enginelocation	wheelbase	carlength	carwidth	carheight	curbweight	enginetype	...	vw rabbit	gas
0	1	3	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	...	0	1
1	2	3	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	...	0	1
2	3	1	rwd	front	94.5	171.2	65.5	52.4	2823	ohcv	...	0	1
3	4	2	fwd	front	99.8	176.6	66.2	54.3	2337	ohc	...	0	1
4	5	2	4wd	front	99.4	176.6	66.4	54.3	2824	ohc	...	0	1

5 rows × 181 columns

In [120]:

```
CarName.drop(['drivewheel'], axis = 1, inplace = True)
CarName.head()
```

Out[120]:

	car_ID	symboling	enginelocation	wheelbase	carlength	carwidth	carheight	curbweight	enginetype	enginesize	...	vw rabbit	gas
0	1	3	front	88.6	168.8	64.1	48.8	2548	dohc	130	...	0	1
1	2	3	front	88.6	168.8	64.1	48.8	2548	dohc	130	...	0	1
2	3	1	front	94.5	171.2	65.5	52.4	2823	ohcv	152	...	0	1
3	4	2	front	99.8	176.6	66.2	54.3	2337	ohc	109	...	0	1
4	5	2	front	99.4	176.6	66.4	54.3	2824	ohc	136	...	0	1

5 rows × 180 columns

In [121]:

```
status = pd.get_dummies(CarName['enginelocation'])
status.head()
```

Out[121]:

	front	rear
0	1	0
1	1	0
2	1	0
3	1	0
4	1	0

In [122]:

```
status = pd.get_dummies(CarName['enginelocation'], drop_first = True)
CarName = pd.concat([CarName, status], axis = 1)
```

```
CarName.head()
```

Out[122]:

	car_ID	symboling	enginelocation	wheelbase	carlength	carwidth	carheight	curbweight	enginetype	enginesize	...	gas	turbo	t
0	1	3	front	88.6	168.8	64.1	48.8	2548	dohc	130	...	1	0	
1	2	3	front	88.6	168.8	64.1	48.8	2548	dohc	130	...	1	0	
2	3	1	front	94.5	171.2	65.5	52.4	2823	ohcv	152	...	1	0	
3	4	2	front	99.8	176.6	66.2	54.3	2337	ohc	109	...	1	0	
4	5	2	front	99.4	176.6	66.4	54.3	2824	ohc	136	...	1	0	

5 rows × 181 columns

In [123]:

```
CarName.drop(['enginelocation'], axis = 1, inplace = True)
CarName.head()
```

Out[123]:

	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight	enginetype	enginesize	fuelsystem	...	gas	turbo	two
0	1	3	88.6	168.8	64.1	48.8	2548	dohc	130	mpfi	...	1	0	1
1	2	3	88.6	168.8	64.1	48.8	2548	dohc	130	mpfi	...	1	0	1
2	3	1	94.5	171.2	65.5	52.4	2823	ohcv	152	mpfi	...	1	0	1
3	4	2	99.8	176.6	66.2	54.3	2337	ohc	109	mpfi	...	1	0	0
4	5	2	99.4	176.6	66.4	54.3	2824	ohc	136	mpfi	...	1	0	0

5 rows × 180 columns

In [124]:

```
status = pd.get_dummies(CarName['enginetype'])
status.head()
```

Out[124]:

	dohc	dohcv	l	ohc	ohcf	ohcv	rotor
0	1	0	0	0	0	0	0
1	1	0	0	0	0	0	0
2	0	0	0	0	0	1	0
3	0	0	0	1	0	0	0
4	0	0	0	1	0	0	0

In [125]:

```
status = pd.get_dummies(CarName['enginetype'], drop_first = True)
CarName = pd.concat([CarName, status], axis = 1)
CarName.head()
```

Out[125]:

	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight	enginetype	enginesize	fuelsystem	...	wagon	fwd	rwd
0	1	3	88.6	168.8	64.1	48.8	2548	dohc	130	mpfi	...	0	0	1
1	2	3	88.6	168.8	64.1	48.8	2548	dohc	130	mpfi	...	0	0	1
2	3	1	94.5	171.2	65.5	52.4	2823	ohcv	152	mpfi	...	0	0	1
3	4	2	99.8	176.6	66.2	54.3	2337	ohc	109	mpfi	...	0	1	0
4	5	2	99.4	176.6	66.4	54.3	2824	ohc	136	mpfi	...	0	0	0

5 rows × 186 columns

In [126]:

```
CarName.drop(['enginetype'], axis = 1, inplace = True)
CarName.head()
```

Out[126]:

	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight	enginesize	fuelsystem	boreratio	...	wagon	fwd	rwd
0	1	3	88.6	168.8	64.1	48.8	2548	130	mpfi	3.47	...	0	0	1
1	2	3	88.6	168.8	64.1	48.8	2548	130	mpfi	3.47	...	0	0	1
2	3	1	94.5	171.2	65.5	52.4	2823	152	mpfi	2.68	...	0	0	1
3	4	2	99.8	176.6	66.2	54.3	2337	109	mpfi	3.19	...	0	1	0
4	5	2	99.4	176.6	66.4	54.3	2824	136	mpfi	3.19	...	0	0	0

5 rows × 185 columns

In [127]:

```
status = pd.get_dummies(CarName['fuelsystem'])
status.head()
```

Out[127]:

	1bbl	2bbl	4bbl	idi	mfi	mpfi	spdi	spfi
0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	1	0	0
2	0	0	0	0	0	1	0	0
3	0	0	0	0	0	1	0	0
4	0	0	0	0	0	1	0	0

In [128]:

```
status = pd.get_dummies(CarName['fuelsystem'], drop_first = True)
CarName = pd.concat([CarName, status], axis = 1)
CarName.head()
```

Out[128]:

	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight	enginesize	fuelsystem	boreratio	...	ohcf	ohcv	rotor
0	1	3	88.6	168.8	64.1	48.8	2548	130	mpfi	3.47	...	0	0	0
1	2	3	88.6	168.8	64.1	48.8	2548	130	mpfi	3.47	...	0	0	0
2	3	1	94.5	171.2	65.5	52.4	2823	152	mpfi	2.68	...	0	1	0
3	4	2	99.8	176.6	66.2	54.3	2337	109	mpfi	3.19	...	0	0	0
4	5	2	99.4	176.6	66.4	54.3	2824	136	mpfi	3.19	...	0	0	0

5 rows × 192 columns

In [129]:

```
CarName.drop(['fuelsystem'], axis = 1, inplace = True)
CarName.head()
```

Out[129]:

	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight	enginesize	boreratio	stroke	...	ohcf	ohcv	rotor	2bb
0	1	3	88.6	168.8	64.1	48.8	2548	130	3.47	2.68	...	0	0	0	0

	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight	enginesize	boreratio	stroke	...	ohcf	ohcv	rotor	2bb
2	3	1	94.5	171.2	65.5	52.4	2823	152	2.68	3.47	...	0	1	0	0
3	4	2	99.8	176.6	66.2	54.3	2337	109	3.19	3.40	...	0	0	0	0
4	5	2	99.4	176.6	66.4	54.3	2824	136	3.19	3.40	...	0	0	0	0

5 rows × 191 columns

In [130]:

```
from sklearn.model_selection import train_test_split
np.random.seed(0)
df_train, df_test = train_test_split(CarName, train_size = 0.7, test_size = 0.3, random_state = 100
)
```

In [131]:

```
from sklearn.preprocessing import MinMaxScaler
```

In [132]:

```
num_vars = ['wheelbase', 'carlength', 'carwidth', 'carheight', 'curbweight','enginesize']
df_train[num_vars] = scaler.fit_transform(df_train[num_vars])
```

In [133]:

```
df_train.head()
```

Out[133]:

	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight	enginesize	boreratio	stroke	...	ohcf	ohcv	rotor	2
122	123	1	-0.811836	-0.487238	-0.924500	-1.134628	-0.642128	-0.660242	2.97	3.23	...	0	0	0	
125	126	3	-0.677177	-0.359789	1.114978	-1.382026	0.439415	0.637806	3.94	3.11	...	0	0	0	
166	167	1	-0.677177	-0.375720	-0.833856	-0.392434	-0.441296	-0.660242	3.24	3.08	...	0	0	0	
1	2	3	-1.670284	-0.367754	-0.788535	-1.959288	0.015642	0.123485	3.47	2.68	...	0	0	0	
199	200	-1	0.972390	1.225364	0.616439	1.627983	1.137720	0.123485	3.62	3.15	...	0	0	0	

5 rows × 191 columns

In [134]:

```
df_train.describe()
```

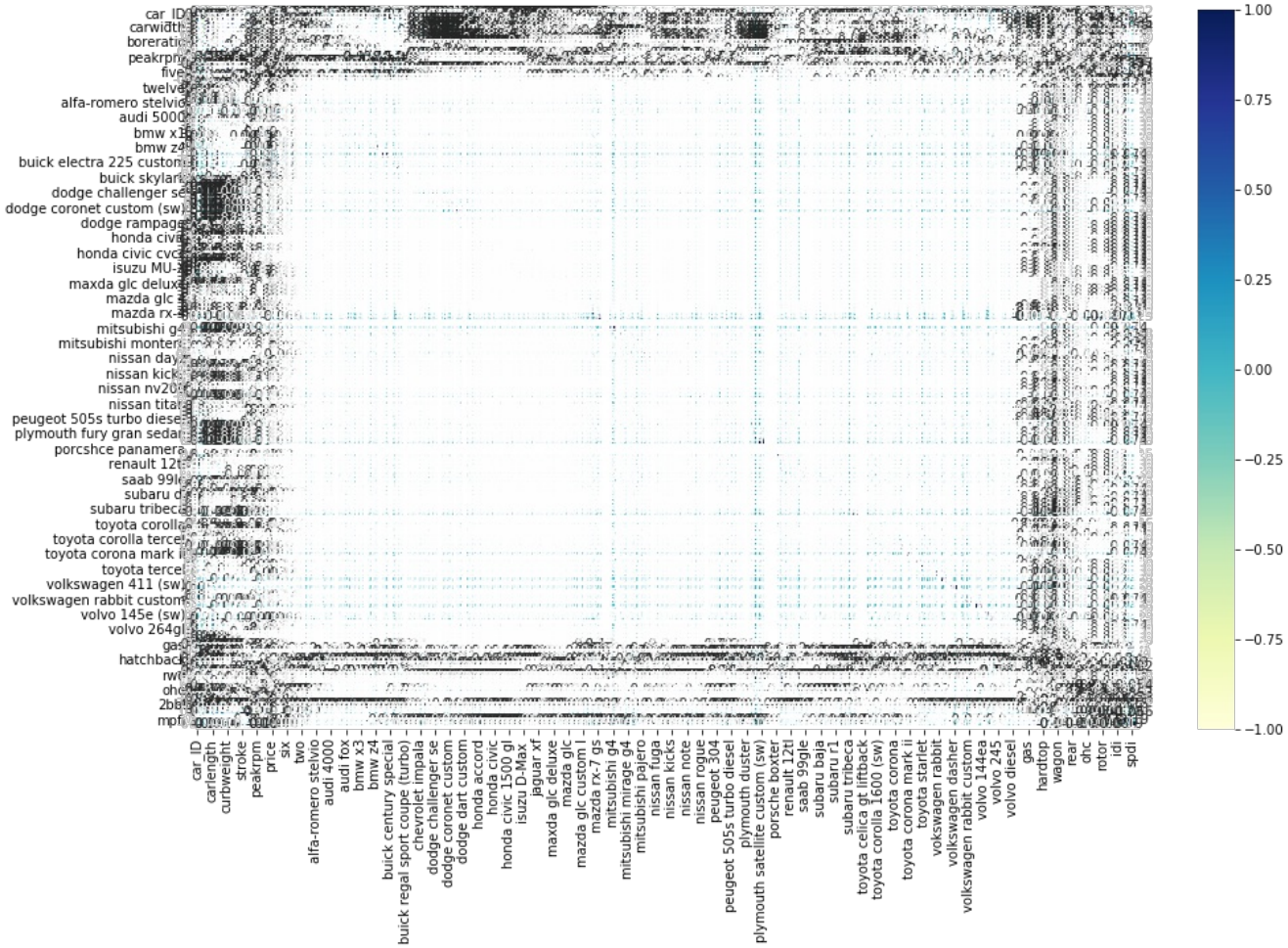
Out[134]:

	car_ID	symboling	wheelbase	carlength	carwidth	carheight	curbweight	enginesize	boreratio	
count	143.000000	143.000000	1.430000e+02	1.430000e+02	1.430000e+02	1.430000e+02	1.430000e+02	1.430000e+02	143.000000	143.
mean	98.524476	0.797203	1.565182e-15	1.614870e-16	-4.074441e-15	5.341493e-16	-1.614870e-16	-6.211038e-17	3.307413	3.
std	58.977655	1.195999	1.003515e+00	1.003515e+00	1.003515e+00	1.003515e+00	1.003515e+00	1.003515e+00	0.260997	0.
min	1.000000	-2.000000	-2.006930e+00	-2.574223e+00	-2.510760e+00	-2.371619e+00	-1.937401e+00	-1.566427e+00	2.680000	2.
25%	48.500000	0.000000	-6.771770e-01	-6.186702e-01	-8.565171e-01	-7.222984e-01	-7.711028e-01	-6.847340e-01	3.065000	3.
50%	97.000000	1.000000	-3.405307e-01	-1.128552e-01	-1.993522e-01	6.112865e-02	-2.478347e-01	-3.663447e-01	3.310000	3.
75%	147.500000	1.000000	4.505882e-01	7.076008e-01	4.804736e-01	7.414732e-01	7.203955e-01	3.928914e-01	3.540000	3.
max	205.000000	3.000000	2.874442e+00	2.324616e+00	2.927846e+00	2.287711e+00	2.812547e+00	4.923816e+00	3.940000	4.

8 rows × 191 columns

In [135]:

```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize = (16, 10))
sns.heatmap(df_train.corr(), annot = True, cmap="YlGnBu")
plt.show()
```



In [136]:

```
y_train = df_train.pop('price')
x_train = df_train
```

In [137]:

```
import statsmodels.api as sm
x_train_lm = sm.add_constant(x_train[['wheelbase']])
lr = sm.OLS(y_train,x_train_lm).fit()
```

In [138]:

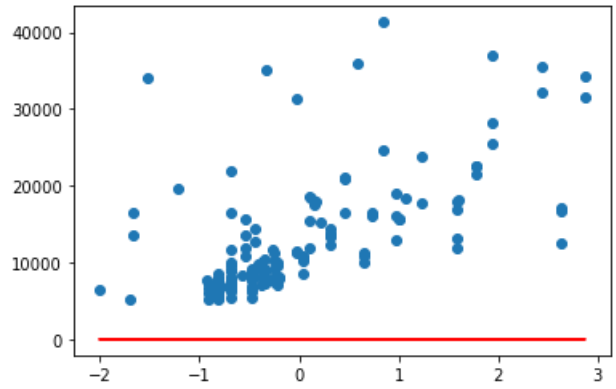
```
lr.params
```

Out[138]:

```
const        13056.347322
wheelbase     4843.563051
dtype: float64
```

In [139]:

```
plt.scatter(x_train_lm.iloc[:, 1], y_train)
plt.plot(x_train_lm.iloc[:, 1],0.127 + 0.462*x_train_lm.iloc[:, 1], 'r')
plt.show()
```



In [140]:

```
print(lr.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.388
Model:                            OLS   Adj. R-squared:                  0.383
Method:                 Least Squares   F-statistic:                     89.25
Date:                Sun, 26 Apr 2020   Prob (F-statistic):           1.03e-16
Time:                        09:16:07   Log-Likelihood:                -1449.0
No. Observations:                 143   AIC:                             2902.
Df Residuals:                     141   BIC:                             2908.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1.306e+04    512.700     25.466      0.000     1.2e+04    1.41e+04
wheelbase   4843.5631    512.700      9.447      0.000    3829.989    5857.137
==============================================================================
Omnibus:                       81.027   Durbin-Watson:                   1.896
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              332.436
Skew:                           2.161   Prob(JB):                     6.49e-73
Kurtosis:                       9.092   Cond. No.                         1.00
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

In [141]:

```
x_train_lm = x_train[['wheelbase', 'carlength']]
```

In [142]:

```
import statsmodels.api as sm
x_train_lm = sm.add_constant(x_train_lm)
lr = sm.OLS(y_train, x_train_lm).fit()
lr.params
```

Out[142]:

```
const        13056.347322
wheelbase     -136.259151
carlength     5672.367489
dtype: float64
```

In [143]:

```
print(lr.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.510
Model:                            OLS   Adj. R-squared:                  0.503
```

```
Model:                            OLS   Adj. R-squared:                  0.505
Method:                 Least Squares   F-statistic:                     72.71
Date:                Sun, 26 Apr 2020   Prob (F-statistic):           2.21e-22
Time:                        09:16:08   Log-Likelihood:                -1433.2
No. Observations:                 143   AIC:                             2872.
Df Residuals:                     140   BIC:                             2881.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1.306e+04    460.484     28.354      0.000    1.21e+04     1.4e+04
wheelbase    -136.2592    961.691     -0.142      0.888   -2037.573    1765.055
carlength    5672.3675    961.691      5.898      0.000    3771.053    7573.682
==============================================================================
Omnibus:                       53.110   Durbin-Watson:                   1.899
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              127.588
Skew:                           1.553   Prob(JB):                     1.97e-28
Kurtosis:                       6.430   Cond. No.                         3.92
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

In [144]:

```
x_train_lm = x_train[['wheelbase', 'carlength','carwidth']]
```

In [145]:

```
import statsmodels.api as sm
x_train_lm = sm.add_constant(x_train_lm)
lr = sm.OLS(y_train,x_train_lm).fit()
lr.params
```

Out[145]:

```
const        13056.347322
wheelbase    -1460.620047
carlength     2068.091646
carwidth      5632.641754
dtype: float64
```

In [146]:

```
print(lr.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.652
Model:                            OLS   Adj. R-squared:                  0.644
Method:                 Least Squares   F-statistic:                     86.66
Date:                Sun, 26 Apr 2020   Prob (F-statistic):           1.14e-31
Time:                        09:16:08   Log-Likelihood:                -1408.7
No. Observations:                 143   AIC:                             2825.
Df Residuals:                     139   BIC:                             2837.
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1.306e+04    389.480     33.522      0.000    1.23e+04    1.38e+04
wheelbase  -1460.6200    832.203     -1.755      0.081   -3106.033     184.793
carlength   2068.0916    943.796      2.191      0.030     202.039    3934.144
carwidth    5632.6418    748.048      7.530      0.000    4153.618    7111.665
==============================================================================
Omnibus:                       65.147   Durbin-Watson:                   1.735
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              245.523
Skew:                           1.687   Prob(JB):                     4.85e-54
Kurtosis:                       8.460   Cond. No.                         4.89
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [147]:

```
CarName.columns
```

Out[147]:

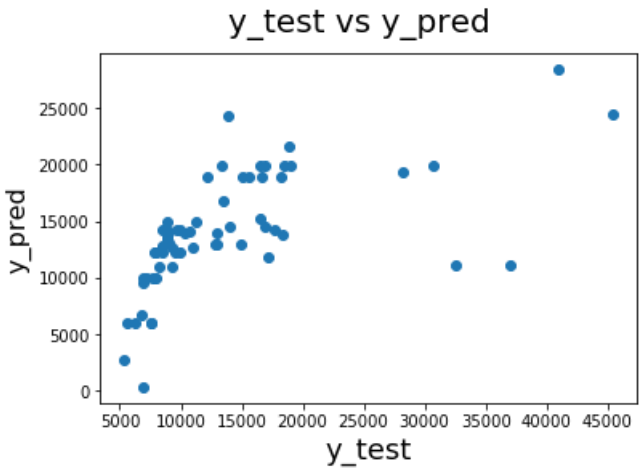
```
Index(['car_ID', 'symboling', 'wheelbase', 'carlength', 'carwidth',
       'carheight', 'curbweight', 'enginesize', 'boreratio', 'stroke',
       ...
       'ohcf', 'ohcv', 'rotor', '2bbl', '4bbl', 'idi', 'mfi', 'mpfi', 'spdi',
       'spfi'],
      dtype='object', length=191)
```

In [159]:

```
fig = plt.figure()
plt.scatter(y_test,y_pred)
fig.suptitle('y_test vs y_pred', fontsize=20)
plt.xlabel('y_test', fontsize=18)
plt.ylabel('y_pred', fontsize=16)
```

Out[159]:

```
Text(0, 0.5, 'y_pred')
```



In [148]:

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

In [153]:

```
def build_model(x,y):
    x= sm.add_constant(x)
    lm = sm.OLS(y,x).fit()
    print(lm.summary())
    return x
def checkVIF(x):
    vif = pd.DataFrame()
    vif['Features'] = x.columns
    vif['VIF'] = [variance_inflation_factor(x.values, i) for i in range(x.shape[1])]
    vif['VIF'] = round(vif['VIF'], 2)
    vif = vif.sort_values(by = "VIF", ascending = False)
    return(vif)
```

In [154]:

```
x_train_new = build_model(x_train_lm,y_train)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.652
```

```
Model:                            OLS   Adj. R-squared:                  0.644
Method:                 Least Squares   F-statistic:                     86.66
Date:                Sun, 26 Apr 2020   Prob (F-statistic):           1.14e-31
Time:                        09:18:31   Log-Likelihood:                -1408.7
No. Observations:                 143   AIC:                             2825.
Df Residuals:                     139   BIC:                             2837.
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1.306e+04    389.480     33.522      0.000    1.23e+04    1.38e+04
wheelbase  -1460.6200    832.203     -1.755      0.081   -3106.033     184.793
carlength   2068.0916    943.796      2.191      0.030     202.039    3934.144
carwidth    5632.6418    748.048      7.530      0.000    4153.618    7111.665
==============================================================================
Omnibus:                       65.147   Durbin-Watson:                   1.735
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              245.523
Skew:                           1.687   Prob(JB):                     4.85e-54
Kurtosis:                       8.460   Cond. No.                         4.89
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

In [155]:

```
x_train_new = build_model(x_train_new,y_train)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.652
Model:                            OLS   Adj. R-squared:                  0.644
Method:                 Least Squares   F-statistic:                     86.66
Date:                Sun, 26 Apr 2020   Prob (F-statistic):           1.14e-31
Time:                        09:18:38   Log-Likelihood:                -1408.7
No. Observations:                 143   AIC:                             2825.
Df Residuals:                     139   BIC:                             2837.
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1.306e+04    389.480     33.522      0.000    1.23e+04    1.38e+04
wheelbase  -1460.6200    832.203     -1.755      0.081   -3106.033     184.793
carlength   2068.0916    943.796      2.191      0.030     202.039    3934.144
carwidth    5632.6418    748.048      7.530      0.000    4153.618    7111.665
==============================================================================
Omnibus:                       65.147   Durbin-Watson:                   1.735
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              245.523
Skew:                           1.687   Prob(JB):                     4.85e-54
Kurtosis:                       8.460   Cond. No.                         4.89
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

In [157]:

```
checkVIF(x_train_new)
```

Out[157]:

	Features	VIF
2	carlength	5.87
1	wheelbase	4.57
3	carwidth	3.69
0	const	1.00

In [ ]: